CITY UNIVERSITY LONDON, INM432 BIG DATA

TILLMAN WEYDE, DANIEL WOLFF, SON N. TRAN

Cousework 2014: Large-Scale Text Classification

This coursework is about analysing the text collection of project Gutenberg (`www.gutenberg.org`). The complete text collection has over 20 GB. We will provide a subset on lewes in the directory `/data/extra/gutenberg/text-part` we also provide the metadata in XML format in the directory `/data/extra/gutenberg/meta`. Once you have a version you are happy with, you can run your programs on the full dataset `/data/extra/gutenberg/text-full`.

The task of this coursework is first to load, parse and process large-scale unstructured and structured data. The next step is the development of classifiers that recognise the subject of a given text. Thirdly it is about evaluating the performance of the classifiers in terms of classification accuracy as well as resource usage and reasoning about the performance as well as possible applications.

This coursework can be done **in pairs** or **individually**. It is highly recommended to work in pairs if feasible for you, and ideally to have one experienced programmer in each pair. If you work in a pair you need to address one additional task (no 5) and marks will be normalised out of 125.

1 Reading and preparing text files (25%)
a) Start by traversing the `text-part` directory , and loading all text files using `loadTextFile()`, which loads the text as lines.
b)From the text files you need to remove the header. The last line of the header starts and ends with `***`.
c)You need to extract the ID of the text, which occurs in the header in the format `[EBook #<ID>]`, where ID is a natural number.
d) Extract the list of Word Frequency pairs per file (as an RDD) and save it to disk for later use.
e) Calculate the IDF values and save the list of (word,IDF) pairs for later use.
f) Calculate the TF.IDF values and create a 10000 dimensional vector per document using the hashing trick.

2 Reading and preparing metadata from XML (15%)
a) Read the metadata files from the `meta` directory as files (not RDDs).
b) Parse the XML (will be covered after Reading week). **c)** Extract the subjects and the ID which is given in `pgterms:ebook` element as the `rdf:about` attribute in the form `ebooks/<ID>`. The subjects are given in the metadata, both in free text and using subject codes according to the Dublin Core standard (mostly in the Library of Congress

Classification).
d) Store the subject lists per file as an RDD for later use.

3 Training classifiers (30%)
a) Find the 10 most frequent subjects.
b) For each of these subjects, build a classifier (contains/does not contain the subject) using:
i) Naive Bayes
ii) Decision trees
iii) Logistic regression
c) Find the best regularisation values for *Decision Trees* and *Logistic Regression* using *Cross Validation* and document the training validation and and testing error for each classification method and each regularisation value.

4 Efficiency, measurements, and discussion. (30%)
a) Experiment with different Vector sizes and document the changes in accuracy, running time and memory usage (will be covered after reading week).
b) Discuss the results of task 4a) and those of task 3) with respect to the known complexities of the used algorithms.
c) Discuss possible application scenarios and for the classification, and what suitable extensions or optimisations in those scenarios are. Possible aspects to consider are: the encoded subject classes, user needs, natural languages, and the related technical requirements.

5 Task for pairs Further improvements. (40%)
a) Double hash vector.
i) Implement a feature vector that is divided into to parts of similar but different size, apply the same hash function, but perform the modulo operation separately for each part.
ii) Compare this version experimentally with the previous version in terms of accuracy.
iii) Document and interpret the results of your test.

b) Estimating the author's year of birth. This is an experimental task to see whether we can find stylistic properties of an English text the enable us to estimate the time of the authors birth.
a) Extract the author's year of birth from the metadata.
b) Train a linear regression model (LassoWithSGD from mllib) on the TF.IDF vectors.
c) Discuss the results and options for improvement.

Submission details Submission is due on 29th Nov via Moodle. Please submit your code together with instructions on how to run it (i.e. the command line). The report with experimental results and textual answers should be no longer than 1500 words and 6 pages A4 (2000 words/8pages for pairs).